Exercise Worksheet

[www.vomtom.at](http://www.vomtom.at)

# From the Course: Understanding Docker Run, Dockerfile, Docker-Compose for Beginners

## Databases and Data Persistence in Named Volumes in Docker Step by Step

docker volume create --name my-vol

* Create a volume called “my-vol”

docker volume ls

* List all volumes
* Observe “my-vol” is present

Now we have to use it somehow. Use the following docker-compose.yml file:



Run

docker-compose up -d

* This will bring up the db-container
* It will write into the volume “my-vol” all the database data

docker run -v my-vol:/mydata --rm -it ubuntu /bin/bash

* Will start a new container with ubuntu
* Will mount my-vol into /mydata in the container

ls /mydata

* Should show the database data files
* This way it’s easy to move volumes around

exit

* Exit the container again

But how about sharing data between two containers? Let’s try:

docker volume create --name Datastore1

* Creates a new volume called “Datastore1”

docker run -v Datastore1:/mydatastore --rm -it ubuntu /bin/bash

* Opens a shell with Datastore1 in /mydatastore

echo “hello datastore1” > /mydatastore/hello.txt

* Writes a new text-file

Open a second terminal!

Cmd2: docker run -v Datastore1:/mydatastore1 --rm -it ubuntu /bin/bash

* Opens a second docker instance
* Connects to the same volume “Datastore1”
* In another directory

Cmd2: cat /mydatastore1/hello.txt

* Should output “hello datastore1”

Cmd2: echo “\n\nhello datastore 2” >> /mydatastore/hello.txt

* Add in another line

Move to the other command line:

Cmd1: cat /mydatastore/hello.txt

* Should now contain both strings

Cmd1: exit

* Exit the first container

Cmd2: exit

* Exit the second container